Chapter 1 Introduction Python

Python on November 24, 2017

Introduction Python

Nguyen Quoc Long



Contents

Definition and notations

Definition Complexity

Formulas

Basic methods for asymptotic behaviour analysis

Counting number of elementary operations

Nguyen Quoc Long Java developer Ikorn Solutions

Contents

Nguyen Quoc Long



Definition and notations

Definition Complexity

Formulas

2 Basic methods for asymptotic behaviour analysis

Counting number of elementary operations

Definition and notations

Definition

Complexity

Formulas

Rasic methods for asymptotic behaviour analysis

Definition

What is a python?

 $Python\ is\ a\ cross-platform\ programming\ language.$

Introduction Python

Nguyen Quoc Long



Contents

Definition and notations

Definition

Complexity

Formulas

Basic methods for asymptotic behaviour analysis

Definition

Nguyen Quoc Long

BK TP.HCM

Introduction Python

What is a python?

Python is a cross-platform programming language.

Properties of python

- Design by Guido van Rossum (1991),
- Paradigm multi-paradigm, object-oriented, imperative, functional, procedural, reflective,
- Typing discipline duck, dynamic, strong,

Contents

Definition and

Definition

Complexity

Formulas

Basic methods for asymptotic behaviour analysis

Definition and notations

Definition

Formulas

Basic methods for

Counting number of

mptotic	behaviou
alysis	

elementary operations

•	Generally,	not	much	interested	in	time	and	space	comp	lexity	for
	small inpu	ıts.									

• Given two algorithms \underline{A} and \underline{B} for solving problem P.

Input size	Algorithm A	Algorithm B
n	5000 n	1.2^{n}
10	50,000	6
100	500,000	2,817,975
1,000	5,000,000	1.5×10^{79}
100,000	5×10^{8}	1.3×10^{7918}

Definition Complexity

Formulas

Basic methods for asymptotic behaviour

analysis elementary operations

Counting number of

- B cannot be used for large inputs, while A is still feasible.
- So what is important is the growth of the complexity functions.
- Growth of time and space complexity with increasing input size nis a suitable measure for the comparison of algorithms.

- Generally, not much interested in time and space complexity for small inputs.
- Given two algorithms A and B for solving problem P.

Input size	Algorithm A	Algorithm B
n	5000 n	1.2^{n}
10	50,000	6
100	500,000	2,817,975
1,000	5,000,000	1.5×10^{79}
100,000	5×10^{8}	1.3×10^{7918}

• Exact formulas, e.g., C(n) = n(n-1)/2.

Introduction Python

Nguyen Quoc Long



Contents

Definition and notations

Definition

Complexity

Basic methods for asymptotic behaviour analysis

Introduction Python Nguyen Quoc Long



Contents

Definition and notations

Definition Complexity

Formulas

Basic methods for asymptotic behaviour analysis

- Exact formulas, e.g., C(n) = n(n-1)/2.
- · Formula indicating order of growth with specific multiplicative constant e.g., $C(n) \approx 0.5n^2$.

Introduction Python

Nguyen Quoc Long



• Exact formulas, e.g., C(n) = n(n-1)/2.

- Formula indicating order of growth with specific multiplicative constant e.g., C(n) ≈ 0.5n².
- Formula indicating order of growth with unknown multiplicative constant e.g., $C(n) \approx c.n^2$

Contents

Definition and notations

Definition Complexity

Formulas

Basic methods for asymptotic behaviour analysis

Introduction Python

Nguyen Quoc Long



Contents

Definition and notations

Definition Complexity

Formulas

Basic methods for asymptotic behaviour analysis

- Exact formulas, e.g., C(n) = n(n-1)/2.
- Formula indicating order of growth with specific multiplicative constant e.g., C(n) ≈ 0.5n².
- Formula indicating order of growth with unknown multiplicative constant e.g., $C(n) \approx c.n^2$
- Most important: Order of growth within a constant multiple as $n \to \infty$

• Exact formulas, e.g., C(n) = n(n-1)/2.

constant e.g., $C(n) \approx 0.5n^2$.

constant e.g., $C(n) \approx c.n^2$



Contents

Definition and notations

Definition Complexity

Formulas

Rasic methods for asymptotic behaviour analysis

elementary operations

Counting number of

Asymptotic growth rate

 $n \to \infty$

A way of comparing functions that ignores constant factors and small input sizes

Formula indicating order of growth with specific multiplicative

Formula indicating order of growth with unknown multiplicative

Most important: Order of growth within a constant multiple as

- O(g(n)): class of functions f(n) that grow no faster than g(n)
- $\Theta(q(n))$: class of functions f(n) that grow at the same rate as g(n)
- $\Omega(g(n))$: class of functions f(n) that grow at least as fast as g(n)

Complexity classes - a small vocabulary

Introduction Python Nguyen Quoc Long

Contents

Definition and notations

Definition

Complexity

Formulas

Basic methods for asymptotic behaviour analysis

Counting number of

elementary operations

- Constant: O(1) (independing on the input size)
- Sub-linear or logarithmic: $O(\log n)$
- Linear: O(n)
- Quasi-linear: $O(n \log n)$
- Quadratic: $O(n^2)$
- Cubic: $O(n^3)$
- Polynomial: $O(n^p)$ ($O(n^2)$, $O(n^3)$, etc)
- Quasi-polynomial: $O(n^{\log(n)})$
- Exponential: $O(2^n)$
- Factorial: O(n!)

Asymptotic upper bound - worst case

Asymptotic upper bound "big O"

$$T(n) = O(f(n))$$
 iif $\exists c \in R^+$, $c > 0$ and $\exists n_0 \in N$, $n_0 > 0$ such that $\forall n > n_0$: $T(n) \le c \times f(n)$

Introduction Python

Nguyen Quoc Long



Contents

Definition and notations

Definition

Complexity

Formulas

Basic methods for asymptotic behaviour analysis



Asymptotic upper bound "big O"

T(n) = O(f(n)) iif $\exists c \in R^+$, c > 0 and $\exists n_0 \in N$, $n_0 > 0$ such that $\forall n > n_0$: $T(n) \le c \times f(n)$

Example

Let $T(n) = 2n + 3n^3 + 5$. T(n) is in $O(n^3)$ with:

• $(c = 8 \text{ and } n_0 = 1) \text{ or } (c = 5 \text{ and } n_0 = 2)$

Contents

Definition and notations

Definition Complexity

Formulas

Basic methods for asymptotic behaviour analysis



Asymptotic upper bound "big O"

T(n) = O(f(n)) iif $\exists c \in R^+$, c > 0 and $\exists n_0 \in N$, $n_0 > 0$ such that $\forall n > n_0$: $T(n) \le c \times f(n)$

Example

Let $T(n) = 2n + 3n^3 + 5$. T(n) is in $O(n^3)$ with:

• $(c=8 \text{ and } n_0=1) \text{ or } (c=5 \text{ and } n_0=2)$

Principle: the lower-order terms are negligible.

Contents

Definition and

Definition Complexity

Formulas

Basic methods for asymptotic behaviour analysis

Asymtotic lower bound - best case

BK TP.HCM

Introduction Python

Nguyen Quoc Long

"big Omega"

 $T(n)=\Omega(f(n))$ iif $\exists c\in R^+$, c>0 and $\exists n_0\in N$, $n_0>0$ such that $\forall n>n_0\colon T(n)\geq c\times f(n)$

Contents

Definition and notations

Definition Complexity

Formulas

Basic methods for asymptotic behaviour analysis

"big Omega"

 $T(n) = \Omega(f(n))$ iif $\exists c \in R^+$, c > 0 and $\exists n_0 \in N$, $n_0 > 0$ such that $\forall n > n_0$: $T(n) \ge c \times f(n)$

Example

Let $T(n) = 2n + 3n^3 + 5$. T(n) is in $\Omega(n^3)$ with:

• $(c = 1 \text{ and } n_0 = 1)$

Contents

Definition and notations

Complexity

Formulas

Basic methods for asymptotic behaviour analysis

Asymtotic approximating bound - average case

"big Theta"

$$\begin{split} T(n) &= \Theta(f(n)) \text{ iif } \exists c_1, c_2 \in R^+, \ c_1 > 0, \ c_2 > 0 \text{ and } \exists n_0 \in N, \\ n_0 &> 0 \\ \text{such that } \forall n > n_0 \colon c_1 \times f(n) \leq T(n) \leq c_2 \times f(n) \end{split}$$

Introduction Python

Nguyen Quoc Long



Contents

Definition and notations

Definition

Complexity

Formulas

Basic methods for asymptotic behaviour analysis

"big Theta"

$$\begin{split} T(n) &= \Theta(f(n)) \text{ iif } \exists c_1, c_2 \in R^+, \ c_1 > 0, \ c_2 > 0 \text{ and } \exists n_0 \in N, \\ n_0 &> 0 \\ \text{such that } \forall n > n_0 \colon c_1 \times f(n) \leq T(n) \leq c_2 \times f(n) \end{split}$$

Property

$$T(n) = O(f(n))$$
 and $T(n) = \Omega(f(n)) \Longrightarrow T(n) = \Theta(f(n))$

Contents

Definition and notations

Definition Complexity

Formulas

Basic methods for asymptotic behaviour analysis

$$T(n) = \Theta(f(n))$$
 iif $\exists c_1, c_2 \in R^+$, $c_1 > 0$, $c_2 > 0$ and $\exists n_0 \in N$, $n_0 > 0$ such that $\forall n > n_0$: $c_1 \times f(n) < T(n) < c_2 \times f(n)$

Property

$$T(n) = O(f(n))$$
 and $T(n) = \Omega(f(n)) \Longrightarrow T(n) = \Theta(f(n))$

Example

Let $T(n)=2n+3n^3+5$. So, T(n) is in $O(n^3)$ and in $\Omega(n^3)$. Consequently, T(n) is in $\Theta(n^3)$.

Contents

Definition and notations

Definition Complexity

Formulas

Basic methods for asymptotic behaviour analysis

BK

Contents

Definition and notations

Definition

Complexity

Basic methods for asymptotic behaviour analysis

- $f(n) = n^2$; g(n) = n
- $\Rightarrow f(n) = O(n^2) = g(n)$ but $f(n) \neq g(n)$

Contents

Definition and notations

Definition

Basic methods for asymptotic behaviour analysis

Complexity

Formulas

Counting number of

elementary operations

Not transitive

- $f(n) = n^2$; g(n) = n
- $\Rightarrow f(n) = O(n^2) = g(n)$ but $f(n) \neq g(n)$

Transitivity

- $f(n) = O(q(n)) \& q(n) = O(h(n)) \Rightarrow f(n) = O(h(n))$
- $f(n) = \Omega(g(n)) \& g(n) = \Omega(h(n)) \Rightarrow f(n) = \Omega(h(n))$
- $f(n) = \Theta(q(n)) \& q(n) = \Theta(h(n)) \Rightarrow f(n) = \Theta(h(n))$

Not transitive

- $f(n) = n^2$; g(n) = n
- $\bullet \ \Rightarrow f(n) = O(n^2) = g(n) \text{ but } f(n) \neq g(n)$

Transitivity

- f(n) = O(g(n)) & $g(n) = O(h(n)) \Rightarrow f(n) = O(h(n))$
- $f(n) = \Omega(g(n))$ & $g(n) = \Omega(h(n)) \Rightarrow f(n) = \Omega(h(n))$
- $f(n) = \Theta(g(n))$ & $g(n) = \Theta(h(n)) \Rightarrow f(n) = \Theta(h(n))$

Additivity

- f(n) = O(h(n)) & $g(n) = O(h(n)) \Rightarrow f(n) + g(n) = O(h(n))$
- $f(n) = \Omega(h(n))$ & $g(n) = \Omega(h(n)) \Rightarrow f(n) + g(n) = \Omega(h(n))$
- $f(n) = \Theta(h(n))$ & $g(n) = \Theta(h(n)) \Rightarrow f(n) + g(n) = \Theta(h(n))$

BK TP.HCM

Contents

Definition and notations

Complexity

ormulas

Basic methods for asymptotic behaviour analysis

Exercise

Introduction Python

Nguyen Quoc Long



Contents

Definition and notations

Definition

Complexity

Formulas

Basic methods for asymptotic behaviour analysis

Contents

Definition and notations

Definition

Complexity

Formulas

Basic methods for asymptotic behaviour analysis

Exercise

Introduction Python

Nguyen Quoc Long



Contents

Definition and notations

Definition Complexity

Formulas

Basic methods for asymptotic behaviour analysis

•
$$T(n) = 3 + 5n^2 \Rightarrow T(n) = \Theta(n^2)$$
 ?

• if
$$T(n)=\left\{ egin{array}{ll} 2n+5 & if & n & \text{is even} \\ n^2-n+1 & if & n & \text{is odd} \end{array} \right.$$
 , then $T(n)=O(?)$ and $T(n)=\Omega(?)$.

Exercise

Compare the asymptotic behaviours of

- 1 2^n and 10^n
- $2 \log_2 n$ and $\log_{10} n$

Introduction Python

Nguyen Quoc Long



Contents

Definition and notations

Definition

Complexity

Formulas

Basic methods for asymptotic behaviour analysis

- 1 2^n and 10^n
- $\log_2 n$ and $\log_{10} n$

- 1 Prove that for any positive functions f and g, f(n) + g(n) and max(f(n);g(n)) are asymptotically equivalent.
- **2** Give a (necessary and sufficient) condition on positive functions f and g to ensure that f(n) + g(n) and f(n) are asymptotically equivalent.

Contents

Definition and notations

Definition Complexity

Formulas

Basic methods for asymptotic behaviour analysis

Common asymptotic behaviours

Introduction Python

Nguyen Quoc Long



Contents

Definition and notations

Definition Complexity

Formulas

Basic methods for asymptotic behaviour analysis

Size	Approximate computational time						
n	$\Theta(\log n)$	$\Theta(n)$	$\Theta(n \log n)$	$\Theta(n^2)$	$\Theta(2^n)$	$\Theta(n!)$	
10	3.10^{-9} s	10^{-8} s	3.10^{-8} s	10^{-7} s	10^{-6} s	3.10^{-3} s	
10^{2}	7.10^{-9} s	$10^{-7} s$	7.10^{-7} s	10^{-5} s	4.10^{13} y	*	
10^{3}	10^{-8} s	10^{-6} s	10^{-5} s	10^{-3} s	*	*	
10^{4}	$1,3.10^{-8}$ s	10^{-5} s	10^{-4} s	$10^{-1} s$	*	*	
10^{5}	$1,7.10^{-8}$ s	$10^{-4} {\rm s}$	2.10^{-3} s	10s	*	*	
10^{6}	2.10^{-8} s	10^{-3} s	2.10^{-2} s	17m	*	*	

Nguyen Quoc Long



Contents

Definition and

notations

Definition Complexity

Formulas

Basic methods for asymptotic behaviour analysis

Counting number of

elementary operations

- 1. Var int: d=0
- **2** For i from 1 to n do
 - 0 d = d + 1
 - $2 a[i] = a[i] \times a[i] + d \times d$
- 3 Endfor

Introduction Python

Nguyen Quoc Long



Contents

Definition and notations

Definition

Complexity Formulas

Basic methods for asymptotic behaviour analysis

Counting number of

elementary operations

- **1)** (1) **Var** int: d = 0
- **2** For i from 1 to n do
 - 0 d = d + 1
 - $a[i] = a[i] \times a[i] + d \times d$
- 3 Endfor

Introduction Python

Nguyen Quoc Long



- **1)** (1) Var int: d = 0
- (n) For i from 1 to n do
 - d = d + 1
 - $a[i] = a[i] \times a[i] + d \times d$
- 3 Endfor

Contents

Definition and notations

Definition

Complexity Formulas

Basic methods for asymptotic behaviour analysis

Nguyen Quoc Long



Contents

Definition and notations

Definition

Complexity Formulas

Basic methods for asymptotic behaviour analysis

- **1)** (1) Var int: d = 0
- (n) For i from 1 to n do
 - (1)d = d + 1
- 3 Endfor

Nguyen Quoc Long



Contents

Definition and notations

Definition

Complexity Formulas

Basic methods for asymptotic behaviour analysis

- **1)** (1) Var int: d = 0
- (n) For i from 1 to n do
 - (1)d = d + 1
 - $(1)a[i] = a[i] \times a[i] + d \times d$
- 3 Endfor

- **1)** (1) Var int: d = 0
- (n) For i from 1 to n do
 - (1)d = d + 1
 - **2** $(1)a[i] = a[i] \times a[i] + d \times d$
- 3 Endfor

Contents

Definition and notations

Definition

Complexity

Formulas

Basic methods for asymptotic behaviour

analysis

Counting number of

Counting number of elementary operations

Number of elementary operations: $1 + n \times (1 + 1) = 2n + 1$.

Linear loop example

Introduction Python

Nguyen Quoc Long



- Contents
- Definition and notations
- Definition
- Complexity Formulas
- Basic methods for asymptotic behaviour analysis

- **1. Var** int: i = 1
- $\textbf{2)} \ \ \mathsf{While} \ i \leq n \ \mathsf{do}$
 - 1. Write "Bonjour"
 - 2 i = i + 1
- 3 EndWhile

2 While $i \leq n$ do

1. Write "Bonjour"

2 i = i + 1

3 EndWhile

- 1. Var int: i = n
- 2 While i > 1 do
 - 1. Write "Bonjour"
 - i = i 1
- 3 EndWhile

Contents

Definition and notations

Definition Complexity

Formulas

Basic methods for asymptotic behaviour analysis

2) While $i \leq n$ do

1 Write "Bonjour"

2 i = i + 1

3 EndWhile

- 1 Var int: i = n
- 2 While i > 1 do
 - Write "Bonjour"
 - 2 i = i 1
- 6 EndWhile

Contents

Definition and notations

Definition Complexity

Formulas Basic methods for asymptotic behaviour

analysis Counting number of

elementary operations

Number of elementary operations: 2n + 1.

Logarithmic loop example

- Introduction Python
- Nguyen Quoc Long



- Contents
- Definition and notations
- Definition
- Complexity Formulas
- Basic methods for asymptotic behaviour
- analysis

 Counting number of
- Counting number of elementary operations

- 1) Var int: i = 1
- **2** While $i \leq n$ do
 - 1. Write "Bonjour"
 - $i = i \times 2$
- 3 EndWhile

- **2** While $i \leq n$ do
 - 1. Write "Bonjour"
 - $i = i \times 2$
- 3 EndWhile

- 1. Var int: i = n
- 2 While i > 1 do
 - 1 Write "Bonjour"
 - 2 i = i/2
- 3 EndWhile

Definition and notations

Definition Complexity

Formulas

Basic methods for asymptotic behaviour analysis

- **1 Var** int: i = 1
- 2 While $i \leq n$ do
 - 1 Write "Bonjour"
 - $2i = i \times 2$
- 3 EndWhile

- 1. Var int: i = n
- 2 While $i \geq 1$ do
 - ① Write "Bonjour"
 - i = i/2
- 3 EndWhile

Definition and

Definition Complexity

Formulas

Basic methods for asymptotic behaviour analysis

Counting number of elementary operations

Number of elementary operations: $1 + \log_2(n)$.

Nested loop example

Nb of iterations = nb of iterations of external loop \times nb of iterations of internal loop

Introduction Python

Nguyen Quoc Long



Contents

Definition and notations

Definition Complexity

Formulas

Basic methods for asymptotic behaviour analysis

- \bigcirc Var int: i=1
- 2) While $i \leq n$ do
 - 1 Var int: j=1
 - 2 While $j \leq n$ do
 - 1 Write "Bonjour"
 - **2** $j = j \times 3$
 - 3 EndWhile
 - **4** i = i + 1
- 3 EndWhile

Definition and notations

Definition Complexity

analysis

Formulas Basic methods for asymptotic behaviour

Counting number of elementary operations

Ch.1. p.17/19

- 1 Var int: i = 1
- 2 While $i \leq n$ do
 - 1) Var int: j=1
 - 2 While $j \leq n$ do
 - 1. Write "Bonjour"
 - $2 j = j \times 3$
 - 3 EndWhile
 - 4. i = i + 1
- 3 EndWhile

Number of elementary operations: $1 + n + n \times \log_3(n)$.

Contents

Definition and notations

Complexity

Basic methods for asymptotic behaviour analysis

- 1 Var int: i
- **2** For i from 1 to n do
 - 1. Var int: t = a[i]
 - 2 Var int: j
 - **3** For j from i-1 to 0 do
 - a[j+1] = a[j]
 - 4. EndFor
 - **5.** a[j+1] = t
- 6 EndFor

Definition and notations

Definition

Complexity Formulas

Rasic methods for asymptotic behaviour analysis

Counting number of

Definition

Formulas

Basic methods for asymptotic behaviour analysis

elementary operations

Counting number of

- (1)Var int: i
- **2** For i from 1 to n do
 - 1. Var int: t = a[i]
 - 2 Var int: j
 - **3** For j from i-1 to 0 do
 - a[j+1] = a[j]
 - 4. EndFor
 - **5.** a[j+1] = t
- 6 EndFor



Definition and notations

Definition

Complexity Formulas

Basic methods for asymptotic behaviour

elementary operations

- (1)Var int: i
- 2) (n) For i from 1 to n do
 - 1. Var int: t = a[i]
 - 2 Var int: j
 - 3 For i from i-1 to 0 do
 - a[j+1] = a[j]
 - 4. EndFor
 - **5.** a[j+1] = t
- 6 EndFor



Definition and notations

Definition

Formulas

Basic methods for asymptotic behaviour analysis

Counting number of

elementary operations

- (1)Var int: i
- 2) (n) For i from 1 to n do
 - (1) Var int: t = a[i]
 - 2 Var int: j
 - 3 For i from i-1 to 0 do
 - a[j+1] = a[j]
 - 4. EndFor
 - **5** a[j+1] = t
- 6 EndFor



Definition and notations

Definition

Complexity Formulas

Basic methods for asymptotic behaviour analysis

- (1)Var int: i
- (n) For i from 1 to n do
 - (1) Var int: t = a[i]
 - (1)Var int: j
 - 3 For i from i-1 to 0 do
 - a[j+1] = a[j]
 - 4. EndFor
 - **5** a[j+1] = t

BK TP.HCM

Contents

Definition and notations

Definition

Complexity Formulas

Basic methods for asymptotic behaviour

analysis

Counting number of

Counting number of elementary operations

- (1)Var int: i
- 2 (n)For i from 1 to n do
 - (1) Var int: t = a[i]
 - (1)Var int: j
 - **3** (?) **For** j from i 1 to 0 **do**
 - a[j+1] = a[j]
 - 4. EndFor
 - **6.** a[j+1] = t
- 3 EndFor

Definition and notations

Definition

Complexity Formulas

Basic methods for asymptotic behaviour analysis

Counting number of

elementary operations

- (1)Var int: i
- (n) For i from 1 to n do
 - (1) Var int: t = a[i]
 - (1)Var int: j
 - **3** (?) For i from i-1 to 0 do
 - (1)a[j+1] = a[j]
 - 4. EndFor
 - **5** a[j+1] = t
- 6 EndFor



- (1) Var int: i
- (n) For i from 1 to n do

Function XYZ(array: a[])

- 1) (1) Var int: t = a[i]
- (1)Var int: j
- **3** (?) **For** j from i 1 to 0 **do**
 - (1)a[j+1] = a[j]
- 4. EndFor
- **5** (1)a[j+1] = t
- 3 EndFor

Contents

Definition and notations

Definition

Complexity Formulas

Basic methods for asymptotic behaviour analysis

Homeworks

Introduction Python Nguyen Quoc Long



Contents

Definition and notations

Definition Complexity

analysis

Formulas

Basic methods for asymptotic behaviour

Counting number of

Give algorithms having number of elementary operations as below.

- $T_1(n) = 3 + 5n$
- $T_2(n) = n \log_2 n$
- $T_3(n) = n^3$
- $T_4(n) = (3n)!$
- $T_5(n) = \log_2(3n)$
- $T_6(n) = 2\log_3(2n)$
- $T_7(n) = n^2 \log_4 n$
- $T_8(n) = \sqrt{n}$
- $T_9(n) = \sqrt[3]{n^2}$
- $T_{10}(n) = 2^n$
- $T_{11}(n) = n!$